



# Nutanix Cryptographic Module for BoringSSL

## FIPS 140-2 Non-Proprietary Security Policy

Document Version 1.2

April 12, 2021

**Prepared for:**



**Nutanix, Inc.**  
1740 Technology Drive, Suite 150  
San Jose, CA 95110  
[nutanix.com](http://nutanix.com)  
+1 855.NUTANIX

**Prepared by:**



**KeyPair Consulting Inc.**  
987 Osos Street  
San Luis Obispo, CA 93401  
[keypair.us](http://keypair.us)  
+1 805.316.5024

## References

Ref	Full Specification Name	Date
[140]	<a href="#">FIPS 140-2, Security Requirements for Cryptographic Modules</a>	12/3/2002
[140AA]	<a href="#">FIPS 140-2 Annex A: Approved Security Functions</a>	6/10/2019
[140AC]	<a href="#">FIPS 140-2 Annex C: Approved Random Number Generators</a>	6/10/2019
[140AD]	<a href="#">FIPS 140-2 Annex D: Approved Key Establishment Techniques</a>	8/12/2020
[140DTR]	<a href="#">FIPS 140-2 Derived Test Requirements</a>	1/4/2011
[140IG]	<a href="#">Implementation Guidance for FIPS 140-2 and the Cryptographic Module Validation Program</a>	1/5/2021
[SP 800-38A]	<a href="#">NIST SP 800-38A, Recommendation for Block Cipher Modes of Operation: Methods and Techniques</a>	12/1/2001
[SP 800-38D]	<a href="#">NIST SP 800-38D, Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC</a>	11/28/2007
[SP 800-38F]	<a href="#">NIST SP 800-38F, Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping</a>	12/13/2012
[SP 800-56Ar1]	<a href="#">NIST SP 800-56A, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography (Revised)</a>	3/14/2007
[SP 800-57P1r5]	<a href="#">NIST SP 800-57 Part 1 Rev. 5, Recommendation for Key Management: Part 1 - General</a>	5/4/2020
[SP 800-67r2]	<a href="#">NIST SP 800-67 Rev. 2, Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher</a>	11/17/2017
[SP 800-90Ar1]	<a href="#">NIST SP 800-90A Rev. 1, Recommendation for Random Number Generation Using Deterministic Random Bit Generators</a>	6/24/2015
[SP 800-131Ar2]	<a href="#">SP 800-131A Rev. 2, Transitioning the Use of Cryptographic Algorithms and Key Lengths</a>	3/21/2019
[SP 800-133r2]	<a href="#">NIST SP 800-133 Rev. 2, Recommendation for Cryptographic Key Generation</a>	6/4/2020
[SP 800-135r1]	<a href="#">NIST SP 800-135 Rev. 1, Recommendation for Existing Application-Specific Key Derivation Functions</a>	12/23/2011
[FIPS 180]	<a href="#">FIPS 180-4, Secure Hash Standard (SHS)</a>	8/4/2015
[FIPS 186]	<a href="#">FIPS 186-4, Digital Signature Standard (DSS)</a>	7/19/2013
[FIPS 197]	<a href="#">FIPS 197, Advanced Encryption Standard (AES)</a>	11/26/2001
[FIPS 198]	<a href="#">FIPS 198-1, The Keyed Hash Message Authentication Code (HMAC)</a>	7/16/2008

## Acronyms and Definitions

Term	Definition
AES	Advanced Encryption Standard
API	Application Programming Interface
CAVP	Cryptographic Algorithm Validation Program
CKG	Cryptographic Key Generation
CMVP	Cryptographic Module Validation Program
CO	Cryptographic Officer
CSP	Critical Security Parameter
CVL	Component Validation List
DRBG	Deterministic Random Number Generator
DTR	Derived Test Requirements
ECDSA	Elliptic Curve Digital Signature Algorithm
EC DH	Elliptic Curve Diffie-Hellman
FIPS	Federal Information Processing Standard
GPC	General Purpose Computer
HMAC	Keyed-Hash Message Authentication Code

Term	Definition
IG	Implementation Guidance
IV	Initialization Vector
KAS	Key Agreement Scheme
KAT	Known Answer Test
KDF	Key Derivation Function
KTS	Key Transport Scheme
KW	Key Wrap
NDRNG	Non-Deterministic Random Number Generator
NIST	National Institute of Standards and Technology
OE	Operating Environment
OS	Operating System
PCT	Pairwise Consistency Test
RSA	Rivest, Shamir, Adleman algorithm
SHA/SHS	Secure Hash Algorithm/Standard
SP	Special Publication

## Table of Contents

<b>1</b>	<b>Introduction .....</b>	<b>5</b>
<b>2</b>	<b>FIPS 140-2 Security Levels.....</b>	<b>6</b>
<b>3</b>	<b>Cryptographic Module Specification.....</b>	<b>7</b>
<b>4</b>	<b>Modes of Operation .....</b>	<b>8</b>
<b>5</b>	<b>Ports and Interfaces .....</b>	<b>8</b>
<b>6</b>	<b>Roles, Authentication and Services .....</b>	<b>9</b>
<b>7</b>	<b>Cryptographic Algorithms &amp; Key Management .....</b>	<b>10</b>
7.1	Approved Cryptographic Algorithms.....	10
7.2	Allowed Cryptographic Algorithms .....	11
7.3	Non-Approved Cryptographic Algorithms .....	11
7.4	Cryptographic Key Management .....	12
7.5	Public Keys .....	12
7.6	Key Generation .....	12
7.7	Key Storage .....	13
7.8	Key Zeroization .....	13
<b>8</b>	<b>Self-Tests .....</b>	<b>14</b>
8.1	Power-On Self-Tests .....	14
8.2	Conditional Self-Tests.....	14
<b>9</b>	<b>Guidance and Secure Operation .....</b>	<b>15</b>
9.1	Installation Instructions.....	15
9.2	Secure Operation .....	15
9.2.1	<i>Initialization .....</i>	<i>15</i>
9.2.2	<i>Usage of AES OFB, CFB and CFB8 .....</i>	<i>15</i>
9.2.3	<i>Usage of AES-GCM .....</i>	<i>15</i>
9.2.4	<i>Usage of Triple-DES.....</i>	<i>15</i>
9.2.5	<i>RSA and ECDSA Keys.....</i>	<i>16</i>

# 1 Introduction

This non-proprietary Security Policy for the *Nutanix Cryptographic Module for BoringSSL*, hereafter referred to as the Module, provides an overview of the product and a high-level description of how it meets the overall Level 1 security requirements of FIPS 140-2.

The Module is an open-source, general-purpose cryptographic library which provides FIPS 140-2 approved cryptographic algorithms to serve BoringSSL and other user-space applications. The Module is classified by FIPS 140-2 as a software module, multi-chip standalone module embodiment.

The validated version of the library is 66005f41fbc3529ffe8d007708756720529da20d.

The cryptographic module was tested on the following operational environments on the general-purpose computer (GPC) platforms detailed below:

*Table 1 – Tested Configurations*

#	Operating System	Processor	Platform	Compiler
1	CentOS 7.5	Intel Xeon 4116 (with PAA)	Nutanix NX-3360-G6	Clang v6.0.1
2	CentOS 7.5	Intel Xeon 4116 (without PAA)	Nutanix NX-3360-G6	Clang v6.0.1
3	CentOS 7.9 on Nutanix AHV Hypervisor v7.0	Intel® Xeon® Gold 6234 (with PAA)	Nutanix NX-3060-G7 Appliance	Clang v6.0.1
4	CentOS 7.9 on Nutanix AHV Hypervisor v7.0	Intel® Xeon® Gold 6234 (without PAA)	Nutanix NX-3060-G7 Appliance	Clang v6.0.1
5	Debian Linux 4.9.0 (Rodete)	Intel Xeon E5-2680 (with PAA)		Clang v6.0.1
6	Debian Linux 4.9.0 (Rodete)	Intel Xeon E5-2680 (without PAA)		Clang v6.0.1
7	Ubuntu Linux 18.04	POWER9 (with PAA)		Clang v6.0.1
8	Ubuntu Linux 18.04	POWER9 (without PAA)		Clang v6.0.1

The Module conforms to [140IG] 6.1 *Single Operator Mode and Concurrent Operators*. Each approved operating system manages processes and threads in a logically separated manner. The module's user is considered the owner of the calling application that instantiates the module.

The Module conforms to [140IG] 1.21 *Processor Algorithm Accelerators (PAA) and Processor Algorithm Implementation (PAI)*. The Intel Processor AES-NI functions are identified by [140IG] 1.21 as a known PAA.

As allowed by [140IG] G.5, *Maintaining validation compliance of software or firmware cryptographic modules*, the validation status of the Module is maintained when operated in the following additional operating environment:

- CentOS 7.4

The CMVP makes no statement as to the correct operation of the Module or the security strengths of the generated keys when the specific operational environment is not listed on the validation certificate.

The GPC(s) used during testing met Federal Communications Commission (FCC) Electromagnetic Interference (EMI) and Electromagnetic Compatibility (EMC) requirements for business use as defined by 47 Code of Federal Regulations, Part 15, Subpart B.

## 2 FIPS 140-2 Security Levels

The FIPS 140-2 security levels for the Module are as follows:

Table 2 – Validation Level by FIPS 140-2 Section

Security Requirement	Security Level
<b>Cryptographic Module Specification</b>	1
<b>Cryptographic Module Ports and Interfaces</b>	1
<b>Roles, Services, and Authentication</b>	1
<b>Finite State Model</b>	1
<b>Physical Security</b>	NA
<b>Operational Environment</b>	1
<b>Cryptographic Key Management</b>	1
<b>EMI/EMC</b>	1
<b>Self-Tests</b>	1
<b>Design Assurance</b>	1
<b>Mitigation of Other Attacks</b>	NA
<b>Overall Level</b>	1

[140] Section 4.5 Physical Security is not applicable, as indicated by [140IG] 1.16 *Software Module* and [140IG] G.3 *Partial Validations and Not Applicable Areas of FIPS 140-2*.

The Module does not implement attack mitigations outside the scope of [140], hence [140] Section 4.11 *Mitigation of Other Attacks* is not applicable per [140IG] G.3.

### 3 Cryptographic Module Specification

The module is a software library providing a C-language application program interface (API) for use by other processes that require cryptographic functionality. All operations of the module occur via calls from host applications and their respective internal daemons/processes. As such there are no untrusted services calling the services of the module.

The physical cryptographic boundary is the general-purpose computer on which the module is installed. The logical cryptographic boundary of the BoringCrypto module is a single object file named bcm.o which is statically linked to BoringSSL. The module performs no communications other than with the calling application (the process that invokes the module services) and the host operating system.

Figure 1 shows the logical relationship of the cryptographic module to the other software and hardware components of the computer.

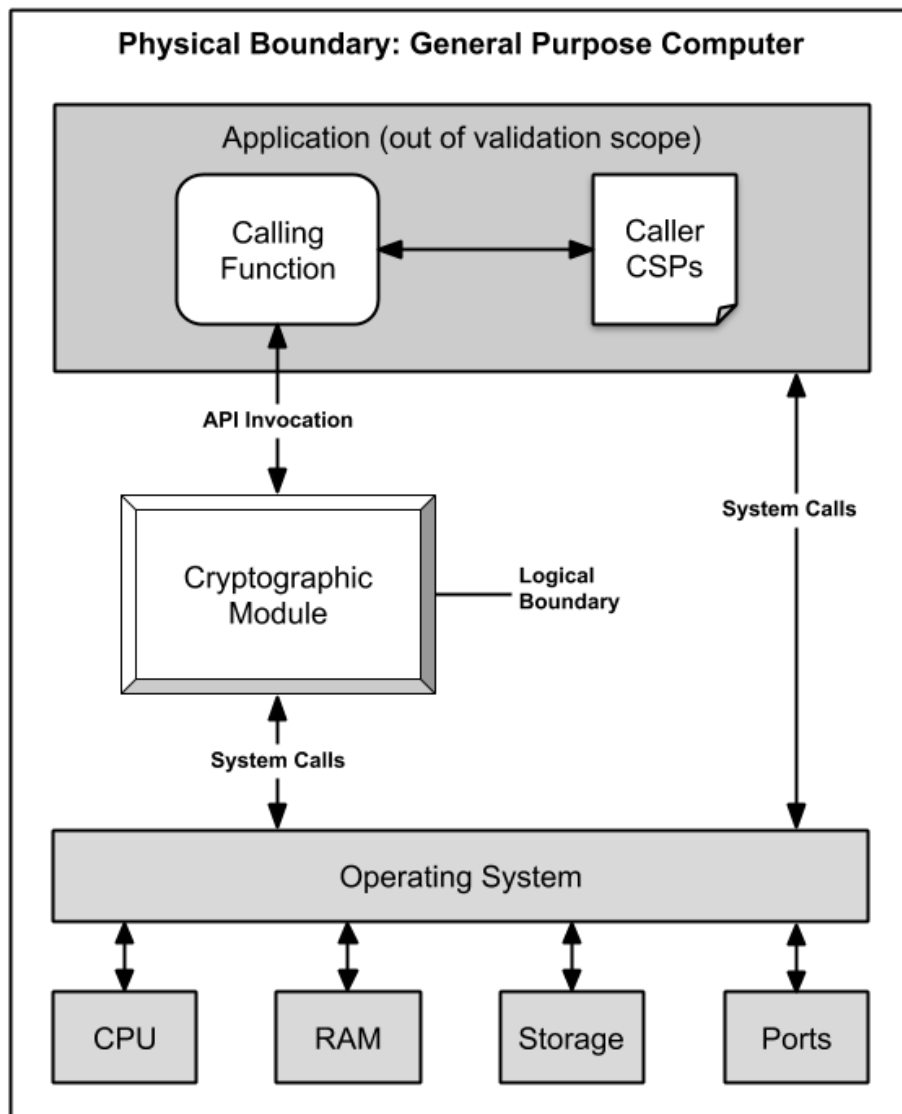


Figure 1 – Logical Boundary

## 4 Modes of Operation

The module supports two modes of operation: Approved and Non-approved. The module will be in FIPS-approved mode when all power up self-tests have completed successfully, and only Approved algorithms are invoked. See Table 7 below for a list of the supported Approved algorithms and Table 8 for allowed algorithms. The non-Approved mode is entered when a non-Approved algorithm is invoked. See Table 9 for a list of non-Approved algorithms.

## 5 Ports and Interfaces

The Data Input interface consists of the input parameters of the API functions. The Data Output interface consists of the output parameters of the API functions. The Control Input interface consists of the actual API input parameters. The Status Output interface includes the return values of the API functions.

*Table 3 – Ports and Interfaces*

FIPS Interface	Physical Ports	Logical Interfaces
<b>Data input</b>	Physical ports of the tested platforms	API input parameters
<b>Data output</b>	Physical ports of the tested platforms	API output parameters and return values
<b>Control input</b>	Physical ports of the tested platforms	API input parameters
<b>Status output</b>	Physical ports of the tested platforms	API return values
<b>Power input</b>	Physical ports of the tested platforms	N/A

As a software module, control of the physical ports is outside module scope; however, when the module is performing self-tests, or is in an error state, all output on the logical data output interface is inhibited.



## 6 Roles, Authentication and Services

The cryptographic module implements both User and Crypto Officer (CO) roles. The module does not support user authentication. The User and CO roles are implicitly assumed by the entity accessing services implemented by the module. A user is considered the owner of the thread that instantiates the module and, therefore, only one concurrent user is allowed.

The Approved services supported by the module and access rights within services accessible over the module's public interface are listed in Table 4.

Table 4 – Approved Services, Roles and Access Rights

Service	Approved Security Functions	Keys and/or CSPs	Roles	Access Rights to Keys and/or CSPs
Module Initialization	N/A	N/A	CO	N/A
Symmetric Encryption/Decryption	AES, Triple-DES	AES, Triple-DES symmetric keys	User, CO	Execute
Keyed Hashing	HMAC-SHA	HMAC key	User, CO	Execute
Hashing	SHS	None	User, CO	N/A
Random Bit Generation	CTR_DRBG	DRBG seed, internal state V and Key values	User, CO	Write/Execute
Signature Generation/Verification	CTR_DRBG, RSA, ECDSA	RSA, ECDSA private key	User, CO	Write/Execute
Key Transport	RSA	RSA private key	User, CO	Write/Execute
Key Agreement	KAS ECC	EC DH private key	User, CO	Write/Execute
Key Generation	CTR_DRBG, RSA, ECDSA	RSA, ECDSA private key	User, CO	Write/Execute
On-demand Self-test	None	None	User, CO	Execute
Zeroization	None	All keys	User, CO	Write/Execute
Show Status	None	None	User, CO	N/A

The module provides the following non-Approved services which utilize algorithms listed in Table 9:

Table 5 – Non-Approved Services

Service	Non-Approved Functions	Roles	Keys and/or CSPs
Symmetric Encryption/Decryption	AES (non-compliant), DES, Triple-DES (non-compliant)	User, CO	N/A
Hashing	MD4, MD5, POLYVAL	User, CO	N/A
Signature Generation/Verification	RSA (non-compliant), ECDSA (non-compliant)	User, CO	N/A
Key Transport	RSA (non-compliant)	User, CO	N/A
Key Generation	RSA (non-compliant), ECDSA (non-compliant)	User, CO	N/A

The module also provides the following non-Approved or non-security relevant services over a non-public interface:

Table 6 – Non-Security Relevant Services

Service	Approved Security Functions	Roles	Access Rights to Keys and/or CSPs
Large integer operations	None	User, CO	N/A
Disable automatic generation of CTR_DRBG "additional_input" parameter	CTR_DRBG	User, CO	N/A
Wegman-Carter hashing with POLYVAL	None	User, CO	N/A

## 7 Cryptographic Algorithms & Key Management

### 7.1 Approved Cryptographic Algorithms

The module implements the following FIPS 140-2 Approved algorithms:

Table 7 – Approved Algorithms and CAVP Certificates

Cert. #	Algorithm	Standard	Mode/Method	Use
<b>5612</b> <b>C1256</b> <b>A1229</b>	AES	[SP 800-38A], [FIPS 197], [SP 800-38D]	128, 192, 256 CBC, ECB, CTR 128, 256 GCM	Encryption, Decryption, Authentication
<b>5612</b> <b>C1256</b> <b>A1229</b>	KTS	[SP 800-38F]	128, 256 AES-KW	Key Wrapping, Key Unwrapping
<b>2035</b> <b>C1256</b> <b>A1229</b>	CVL	[SP 800-135r1]	TLS 1.0/1.1 and 1.2 KDF	Key Derivation
<b>Vendor Affirmed</b>	CKG	[SP 800-133r2]	Cryptographic Key Generation	Key Generation
<b>2253</b> <b>C1256</b> <b>A1229</b>	DRBG	[SP 800-90Ar1]	AES-256 CTR_DRBG	Random Bit Generation
<b>(CVL) 2034</b> <b>1520</b> <b>C1256</b> <b>A1229</b>	ECDSA	[FIPS 186]	SigGen Component KeyPairGen, SigGen, SigVer, PKV P-224, P-256, P-384, P-521	Digital Signature Services
<b>3743</b> <b>C1256</b> <b>A1229</b>	HMAC	[FIPS 198]	HMAC-SHA-1, HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512	Generation, Authentication
<b>(CVL) 2033</b> <b>C1256</b> <b>A1229</b>	KAS ECC	[SP 800-56Ar1]	KAS-ECC Component: Ephemeral Unified	Key agreement scheme
<b>3020</b> <b>C1256</b> <b>A1229</b>	RSA	[FIPS 186]	KeyGen, SigGen, SigVer 1024, 2048, 3072 (Note: Key size 1024 is only used for SigVer)	Digital Signature Services
<b>4509</b> <b>C1256</b> <b>A1229</b>	SHA	[FIPS 180]	SHA-1, SHA-224, SHA-256, SHA-384, SHA-512	Digital Signature Generation, Digital Signature Verification, non-Digital Signature Applications
<b>2825</b> <b>C1256</b> <b>A1229</b>	Triple-DES	[SP 800-38A], [SP 800-67r2]	TCBC, TECB	Encryption, Decryption

## 7.2 Allowed Cryptographic Algorithms

The module supports the following non-FIPS 140-2 Approved but allowed algorithms that may be used in the Approved mode of operation.

*Table 8 – Allowed Algorithms*

Algorithm	Use
<b>EC Diffie-Hellman</b>	CVL Certs. #2033, #2034, #C1256 and #A1229; key agreement; key establishment methodology provides between 112 and 256 bits of encryption strength
<b>RSA Key Transport</b>	Key establishment methodology provides between 112 and 256 bits of encryption strength
<b>MD5</b>	When used with the TLS protocol version 1.0 and 1.1
<b>NDRNG</b>	Used only to seed the Approved DRBG

## 7.3 Non-Approved Cryptographic Algorithms

The module employs the methods listed in Table 9, which are not allowed for use in a FIPS-Approved mode. Their use will result in the module operating in a non-Approved mode.

*Table 9 – Non-Approved Algorithms*

Algorithm	Algorithm
MD5, MD4	DES
AES-GCM (non-compliant)	AES (non-compliant)
ECDSA (non-compliant)	RSA (non-compliant)
POLYVAL	Triple-DES (non-compliant)

## 7.4 Cryptographic Key Management

Table 10 provides a complete list of Private Keys and CSPs used by the module:

*Table 10 – Keys and CSPs Supported*

Key/CSP Name	Key Description	Generated/Input	Output
<b>AES Key</b>	AES (128/192/256) encrypt/decrypt key	Input via API in plaintext	Output via API in plaintext
<b>AES-GCM Key</b>	AES (128/192/256) encrypt/decrypt/generate/verify key	Input via API in plaintext	Output via API in plaintext
<b>AES Wrapping Key</b>	AES (128/192/256) key wrapping key	Input via API in plaintext	Output via API in plaintext
<b>Triple-DES Key</b>	Triple-DES (3-Key) encrypt/decrypt key	Input via API in plaintext	Output via API in plaintext
<b>ECDSA Signing Key</b>	ECDSA (P-224/P-256/P-384/P-521) signature generation key	Internally generated or input via API in plaintext	Output via API in plaintext
<b>EC DH Private Key</b>	EC DH (P-224/P-256/P-384/P-521) private key	Internally generated or input via API in plaintext	Output via API in plaintext
<b>HMAC Key</b>	Keyed hash key (160/224/256/384/512)	Input via API in plaintext	Output via API in plaintext
<b>RSA Key (Key Transport)</b>	RSA (2048 to 16384 bits) key decryption (private key transport) key	Internally generated or input via API in plaintext	Output via API in plaintext
<b>RSA Signature Generation Key</b>	RSA (2048 to 16384 bits) signature generation key	Internally generated or input via API in plaintext	Output via API in plaintext
<b>TLS Master Secret</b>	Shared Secret; 48 bytes of pseudorandom data	Internally derived via key derivation function defined in [SP 800-135r1] KDF (TLS)	Output via API in plaintext
<b>CTR_DRBG V (Seed)</b>	128 bits	Internally generated	Does not exit the module
<b>CTR_DRBG Key</b>	256 bits	Internally generated	Does not exit the module
<b>CTR_DRBG Entropy Input</b>	384 bits	Input via API in plaintext	Does not exit the module

## 7.5 Public Keys

Table 11 provides a complete list of the Public keys used by the module:

*Table 11 – Public Keys Supported*

Public Key Name	Key Description
<b>ECDSA Verification Key</b>	ECDSA (P-224/P-256/P-384/P-521) signature verification key
<b>EC DH Public Key</b>	EC DH (P-224/P-256/P-384/P-521) public key
<b>RSA Key (Key Transport)</b>	RSA (2048 to 16384 bits) key encryption (public key transport) key
<b>RSA Signature Verification Key</b>	RSA (1024 to 16384 bits) signature verification public key

## 7.6 Key Generation

The module supports generation of ECDSA, EC Diffie-Hellman and RSA key pairs as specified in Section 5 of [SP 800-133r2]. The module employs a [SP 800-90Ar1] random bit generator for creation of the seed for asymmetric key generation. The module requests a minimum number of 128 bits of entropy from its Operational Environment per each call.

The output data path is provided by the data interfaces and is logically disconnected from processes performing key generation or zeroization. No key information will be output through the data output interface when the module zeroizes keys.

## **7.7 Key Storage**

The cryptographic module does not perform persistent storage of keys. Keys and CSPs are passed to the module by the calling application. The keys and CSPs are stored in memory in plaintext. Keys and CSPs residing in internally allocated data structures (during the lifetime of an API call) can only be accessed using the module defined API. The operating system protects memory and process space from unauthorized access.

## **7.8 Key Zeroization**

The module is passed keys as part of a function call from a calling application and does not store keys persistently. The calling application is responsible for parameters passed in and out of the module. The Operating System and the calling application are responsible to clean up temporary or ephemeral keys.

## 8 Self-Tests

FIPS 140-2 requires the module to perform self-tests to ensure the integrity of the module and the correctness of the cryptographic functionality at start up. Some functions require conditional tests during normal operation of the module. The supported tests are listed and described in this section.

### 8.1 Power-On Self-Tests

Power-on self-tests are run upon the initialization of the module and do not require operator intervention to run. If any of the tests fail, the module will not initialize. The module will enter an error state and no services can be accessed.

The module implements the following power-on self-tests:

*Table 12 – Power-On Self-Tests*

Type	Test
<b>Integrity Test</b>	HMAC-SHA-512
<b>Known Answer Test (KAT)</b>	AES KAT: Encryption and Decryption. (Key size: 128 bits)
	AES-GCM KAT: Encryption and Decryption. (Key size: 128 bits)
	Triple-DES KAT: Encryption and Decryption. (Key size: 168 bits)
	ECDSA KAT: Signature Generation and Signature Verification. (Curve: P-256)
	HMAC KAT (HMAC-SHA-1, HMAC-SHA-512)
	[SP 800-90Ar1] CTR_DRBG KAT (Key size: 256 bits)
	RSA KAT: Signature Generation and Signature Verification, Encryption and Decryption. (Key size: 2048 bits)
SHA KAT (SHA-1, SHA-256, SHA-512)	

Each module performs all power-on self-tests automatically when the module is initialized. All power-on self-tests must be passed before a User/Crypto Officer can perform services. The Power-on self-tests can be run on demand by power-cycling the host platform.

### 8.2 Conditional Self-Tests

Conditional self-tests are run during operation of the module. If any of these tests fail, the module will enter an error state, where no services can be accessed by the operators. The module can be reinitialized to clear the error and resume FIPS mode of operation. Each module performs the following conditional self-tests:

*Table 13 – Conditional Self-Tests*

Type	Test
<b>Pairwise Consistency Test</b>	ECDSA Key Pair Generation RSA Key Pair Generation
<b>CRNGT</b>	Performed on NDRNG per [140IG] 9.8
<b>DRBG Health Tests</b>	Performed on DRBG, per [SP 800-90Ar1] Section 11.3. Required per [140IG] C.1.

Pairwise consistency tests are performed for both possible modes of use, e.g. Sign/Verify and Encrypt/Decrypt.

## 9 Guidance and Secure Operation

### 9.1 Installation Instructions

During the manufacturing process, Nutanix executes the build and installation instructions for the cryptographic module.

The cryptographic module is pre-installed and configured on supported Nutanix solutions (see Section 1). FIPS mode is enabled by default. There are no additional installation, configuration, or usage instructions for operators intending to use the *Nutanix Cryptographic Module for BoringSSL*.

### 9.2 Secure Operation

#### 9.2.1 Initialization

The cryptographic module is initialized by loading the module before any cryptographic functionality is available. In User Space the operating system is responsible for the initialization process and loading of the library. The module is designed with a default entry point (DEP) which ensures that the power-up tests are initiated automatically when the module is loaded.

#### 9.2.2 Usage of AES OFB, CFB and CFB8

In approved mode, users of the module must not utilize AES OFB, CFB and CFB8.

#### 9.2.3 Usage of AES-GCM

In the case of AES-GCM, the IV generation method is user selectable and the value can be computed in more than one manner.

Following RFC 5288 for TLS, the module ensures that it's strictly increasing and thus cannot repeat. When the IV exhausts the maximum number of possible values for a given session key, the first party, client or server, to encounter this condition may either trigger a handshake to establish a new encryption key in accordance with RFC 5246, or fail. In either case, the module prevents IV duplication and thus enforces the security property.

The module's IV is generated internally by the module's Approved DRBG. The DRBG seed is generated inside the module's physical boundary. The IV is 96 bits in length per [SP 800-38D], Section 8.2.2 and [140IG] A.5 scenario 2.

The selection of the IV construction method is the responsibility of the user of this cryptographic module. In approved mode, users of the module must not utilize GCM with an externally generated IV.

Per [140IG] A.5, in the event module power is lost and restored the consuming application must ensure that any of its AES-GCM keys used for encryption or decryption are re-distributed.

#### 9.2.4 Usage of Triple-DES

In accordance with CMVP [140IG] A.13, when operating in a FIPS approved mode of operation, the same Triple-DES key shall not be used to encrypt more than  $2^{20}$  or  $2^{16}$  64-bit data blocks.

The TLS protocol governs the generation of the respective Triple-DES keys. Please refer to IETF RFC 5246 (TLS) for details relevant to the generation of the individual Triple-DES encryption keys. The user is responsible for ensuring that the module limits the number of encrypted blocks with the same key to no more than  $2^{20}$  when utilized as part of a recognized IETF protocol.

For all other uses of Triple-DES the user is responsible for ensuring that the module limits the number of encrypted blocks with the same key to no more than  $2^{16}$ .

**9.2.5 RSA and ECDSA Keys**

The module allows the use of 1024-bit RSA keys for legacy purposes including signature generation, which is disallowed in the FIPS Approved mode as per [SP 800-131Ar2]. Therefore, the cryptographic operations with the non-approved key sizes will result in the module operating in non-Approved mode implicitly.

Approved algorithms shall not use the keys generated by the module's non-Approved key generation methods.